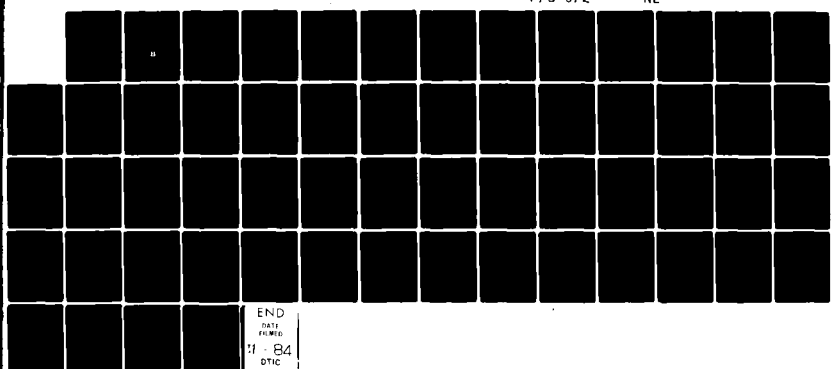
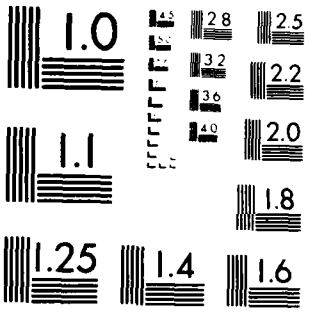
AD-A135 881 HEWLETT-PACKARD 7221T PLOTTER USER'S MANUAL(U) MITRE CORP BEDFORD MA F S DURK NOV 83 MTR-8941 ESD-TR-83-209 F19628-82-C-0001 1/1

UNCLASSIFIED F/G 9/2 NL



END
DATE FILMED
1 - 84
DTIC



ESD-TR-83-209

MTR-8941

AD-A135 881

HEWLETT-PACKARD 7221T PLOTTER USER'S MANUAL

By
F. S. DURK

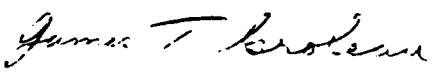
NOVEMBER 1983

Prepared for
DEPUTY FOR TACTICAL SYSTEMS
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
Hanscom Air Force Base, Massachusetts

Project No. 5270
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract No. F19628-82-C-0001

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication.



JAMES T. GROLEAU, GS-13
Electronic Engineer
Correlation & Analysis Division

JOSEPH M. PUFFER, JR., LTC, USAF
Chief, Correlation & Analysis Div
Intelligence Systems Directorate

ANTHONY C. DURANTE, GM-14
Acting Director, Intelligence Sys
Deputy for Tactical Systems

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) MTR-8941 ESD-TR-83-209	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION The MITRE Corporation 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code) Burlington Road Bedford, MA 01730	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Deputy for Tactical Systems 8b. OFFICE SYMBOL (If applicable) TCI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F19628-82-C-0001
8c. ADDRESS (City, State and ZIP Code) Electronic Systems Division, AFSC Hanscom AFB, MA 01731	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. / PROJECT NO. 5270 / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification) HEWLETT-PACKARD 7221T PLOTTER USER'S MANUAL	
12. PERSONAL AUTHOR(S) F. S. Durk	
13a. TYPE OF REPORT Final Report 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day) 1983 November 15. PAGE COUNT 58
16. SUPPLEMENTARY NOTATION	
17. COSATI CODES FIELD / GROUP / SUB GR	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) FORTRAN UTILITY SUBROUTINES PLOTTER HEWLETT-PACKARD 7221T USER MANUAL

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This document is a user manual for the Hewlett-Packard HP7221T plotter. The HP7221T uses a compact binary command language. This manual contains a set of FORTRAN '77 utility subroutines written for a PDP 11/24 which makes it easy to generate plots.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☒ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Susan R. Gilbert	22b. TELEPHONE NUMBER (Include Area Code) (617) 271-8088 22c. OFFICE SYMBOL

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

FOREWORD

This manual documents a set of routines which will prove generally useful to the operation of the HP7221T plotter in the D32 C3CM Computer Lab. It should be considered a beginning rather than a final product. As additional capabilities are required, routines should be added to the library, and the documentation for the routines added to this manual.

A-1

ACKNOWLEDGMENTS

This document was prepared by The MITRE Corporation under Project 5270, Contract F19628-82-C-0001. The contract is sponsored by the Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Massachusetts.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

SECTION 1

INTRODUCTION

GETTING STARTED

This manual describes how to use a set of utility routines (referred to as HPLOT) written in FORTRAN '77 which were developed for the HP7221T plotter using a PDP11/24 with an RSX11M operating system.

The Hewlett-Packard 7221C and 7221T Graphics Plotter Operating and Programming Manual, herein referred to as the HP manual, should be referenced for additional plotter capabilities not accessed by these routines, discussions of the command language required to drive the 7221T, and other techniques for the physical operation of the plotter.

FORTRAN routines are included which will satisfy the most typical program requirements. To add routines to the HPLOT library, follow the procedures discussed in section 4.

To include this library in a program, simply include the object file in the task build process.

UNDERSTANDING GRAPHIC LIMITS AND GRID SIZE

Machine Units

Graphic limits define the desired limits of the plotting area on the platen and are specified in machine units. A machine unit is .025 mm (approximately 0.001 inch) in length, and is the smallest move the plotter can make. The largest possible plotting area is 16000 machine units wide (400 mm) in the x-axis direction by 11400 machine units high (285 mm) in the y-axis direction. These dimensions are the electro-mechanical limits of the plotter and are shown in figure 1-1.

Default Values For Graphic Limits

The plotter assumes the following default graphic limits and advance lengths when roll paper is loaded (paper advance option is "on") and the ENGLISH/METRIC switch at the rear of the plotter is set to ENGLISH.

Lower Left x,y = 520,1020 machine units
Upper Right x,y = 15760,11180 machine units

Half-page length = 8.5 inches
Full-page length = 17 inches

User-Defined Graphic Limits

A user may set his own graphic limits using the front-panel controls or by program control. (Front panel controls are discussed in the HP manual.) A new set of graphic limits permanently replaces any previously established limits.

Plotter Units

Grid size establishes full-scale plotter units, which are the number of units between the point of origin of a plot (0,0) and the full scale values (Xmax,Ymax) of the x- and y-axes. As an example, a grid size of 5 x 7 has 5 plotter units in the x-axis and 7 plotter units in the y-axis. This is shown in figure 1-2. The plotter units establish resolution of a plot, and affect the number of data bytes required to generate a plot. Numerical limits of 16,383 plotter units can be handled in either axis; however, the maximum plotter resolution is .025 mm even when finer resolution is specified by a user.

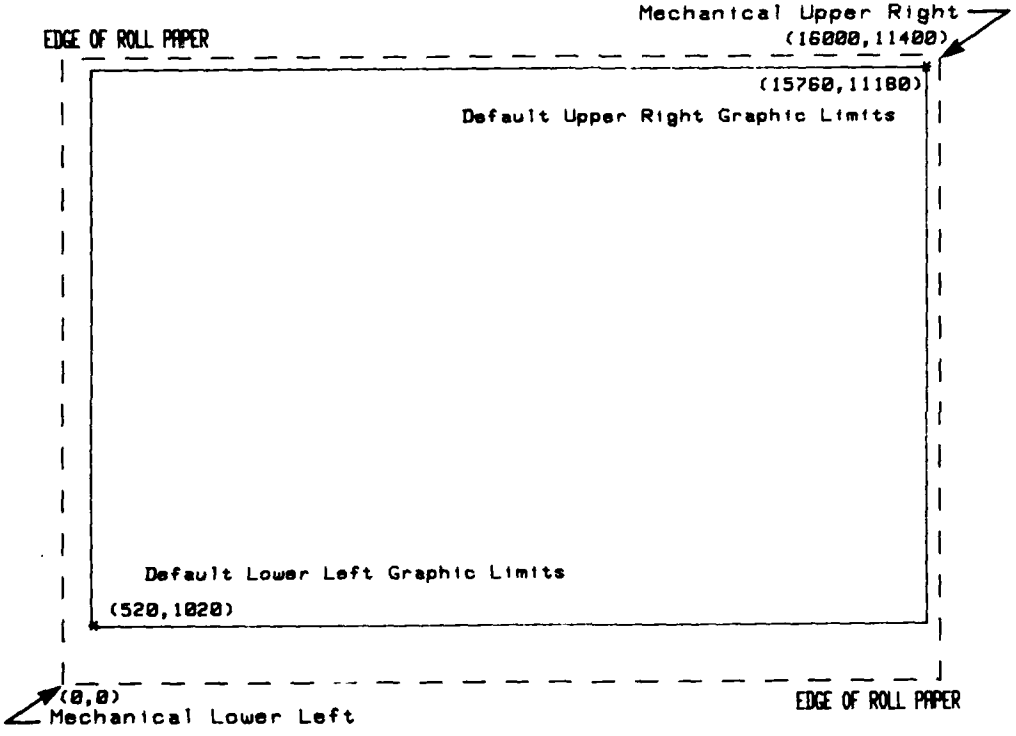


Figure 1-1. Graphic Limits

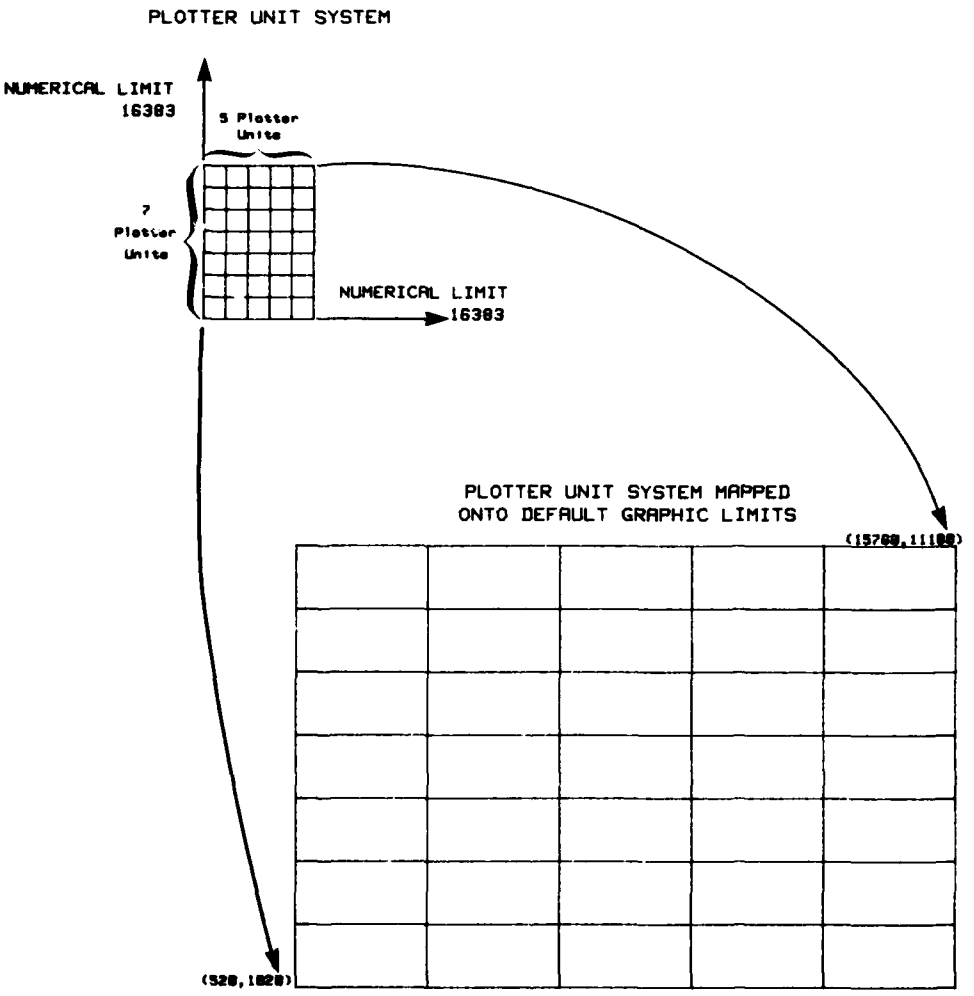


Figure 1-2. Setting a Grid Within Default Graphic Limits

Default Grid Size Values

If a grid size value is not specified in a plotting instruction, the plotter sets the following default grid size:

X-Axis = 3040 plotter units
Y-Axis = 2000 plotter units

When neither graphic limits nor grid size are explicitly specified in a plotter instruction, the 3040 x 2000 grid is mapped onto the previously discussed default graphic limits.

Square Grid Systems

The user sets the grid size by program control. A square grid system is defined when the physical distance on the plot surface corresponding to a single plotter unit is the same in both the x and y directions. For this condition to exist, the aspect ratio of the box established by the current graphic limits must equal the ratio of the grid's full scale x-value to full scale y-value. This may be stated as follows:

Let

X_m = number of machine units in x direction

Y_m = number of machine units in y direction

X_p = number of plotter units in x direction

Y_p = number of plotter units in y direction

Then

$$\left[\frac{X_m}{Y_m} = \frac{X_p}{Y_p}\right] \quad <=> \quad \text{square grid system}$$

Instructions that include angle parameters will produce true angles only if a square grid exists. If a non-square grid exists, arcs will appear elliptical.

The following example (See figure 1-3) illustrates a grid of 5 x 7 established within graphic limits with the same aspect ratio (lower left = 520,1020 machine units and upper right = 5520,8020 machine units).

Default Grid Size Values

If a grid size value is not specified in a plotting instruction, the plotter sets the following default grid size:

X-Axis = 3040 plotter units
Y-Axis = 2000 plotter units

When neither graphic limits nor grid size are explicitly specified in a plotter instruction, the 3040 x 2000 grid is mapped onto the previously discussed default graphic limits.

Square Grid Systems

The user sets the grid size by program control. A square grid system is defined when the physical distance on the plot surface corresponding to a single plotter unit is the same in both the x and y directions. For this condition to exist, the aspect ratio of the box established by the current graphic limits must equal the ratio of the grid's full scale x-value to full scale y-value. This may be stated as follows:

Let

X_m = number of machine units in x direction

Y_m = number of machine units in y direction

X_p = number of plotter units in x direction

Y_p = number of plotter units in y direction

Then

$$\left[\frac{X_m}{Y_m} = \frac{X_p}{Y_p}\right] \quad <=> \quad \text{square grid system}$$

Instructions that include angle parameters will produce true angles only if a square grid exists. If a non-square grid exists, arcs will appear elliptical.

The following example (See figure 1-3) illustrates a grid of 5 x 7 established within graphic limits with the same aspect ratio (lower left = 520,1020 machine units and upper right = 5520,8020 machine units).

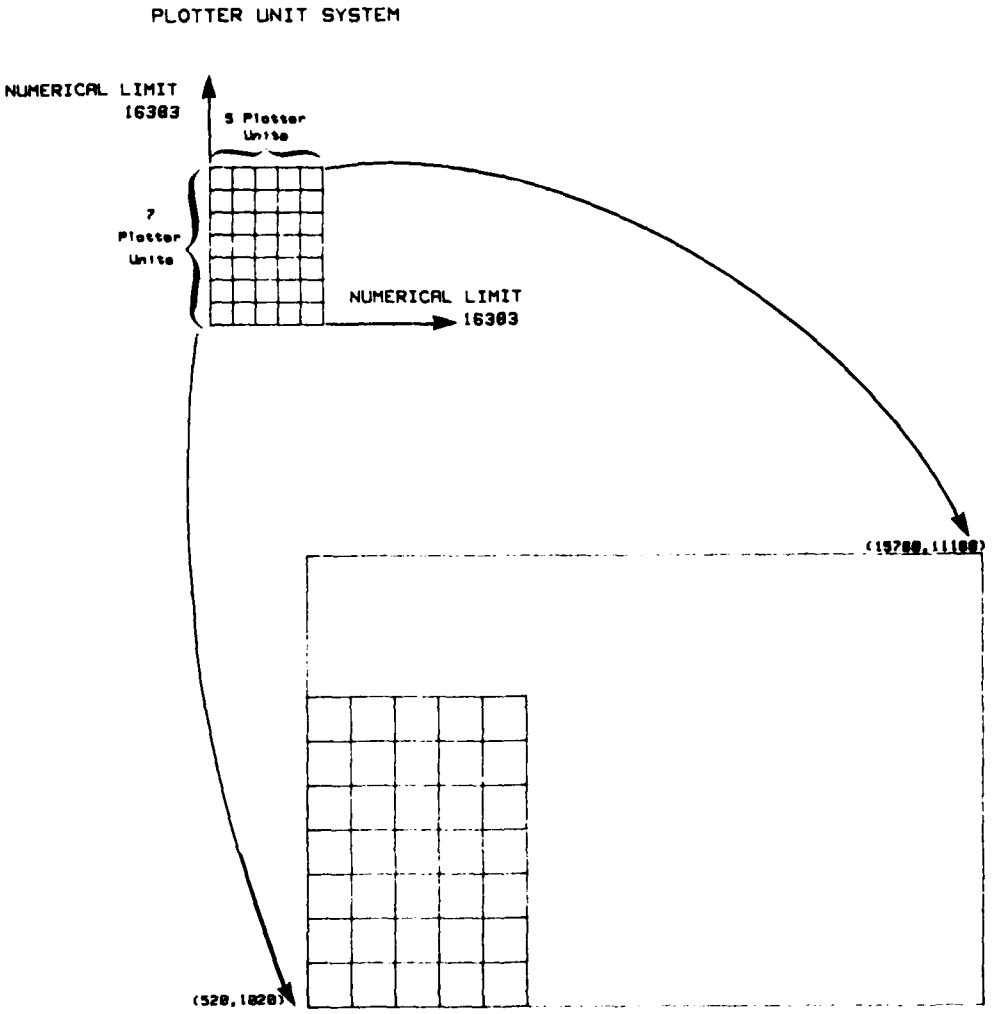


Figure 1-3. Setting a Square Grid

Unit Relationships and Conversions

To determine the number of machine units in a plotter unit:

Length of 1 machine unit = .025 mm (.001 inches)

When the default conditions are used,

$$\frac{\text{X machine units}}{\text{Y plotter units}} = \frac{15760 - 520}{3040} = \frac{15240}{3040} = 5.01 \approx 5$$

$$\frac{\text{X machine units}}{\text{Y plotter units}} = \frac{11180 - 1020}{2000} = \frac{10160}{2000} = 5.08 \approx 5$$

Thus, the default size of one plotter unit in millimeters is:

(number of machine units per plotter unit) x (length of 1 machine unit) =

5 x 0.025 mm = 0.125 mm

Note that the default resolution is one-fifth the maximum attainable resolution. However, integer coordinates with the default grid are so close together (0.125 mm) that pen movement between adjacent coordinates is not discernible.

The following conversions are useful:

1 machine unit = .025 mm

5 machine units = .125 mm

1 inch = 25.4 mm

1 inch = 1016 machine units

SECTION 2

HPLOT ROUTINES

LIST OF ROUTINES BY FUNCTION

Initialize & Terminate Plot

PLOTS	- initialize plotter
PLOTD	- terminate plotter
GRIDSZ	- set grid plotter units
ADVANC	- advance paper

Color

SELPEN	- select pen

Pen Position

MOVE	- move pen
DRAW	- draw vector
IMOVE	- move and then draw incrementally
IDRAW	- draw incrementally

Circles and Arcs

CIRCLE	- draw arc or cirle

Line Type

DASHL	- draw dashed line

Labelling

LABEL	- plot character label
NUMBER	- plot integer value
FNMBER	- plot floating point value
LABSLT	- character slant
LABEND	- alter terminating character of labels
FONT	- change font
LABSIZ	- change size of characters

Rotation

ROTATE	- rotate incremental vectors

ALPHABETICAL LISTING OF ROUTINES

ADVANC(IADV)

Plot paper is advanced and/or cut according to the value of IADV.

Argument	Format	Description
IADV	Integer ϵ[0,4]	0 = Cutter disabled, no page advance 1 = Cutter disabled, full page advance 2 = Cutter disabled, half page advance 3 = Cutter enabled, full page advance 4 = Cutter enabled, half page advance

CIRCLE(IRAD,ANGLE1,ANGLE2)

Draw a circle or an arc starting from the current pen position.

Argument	Format	Description
IRAD	Integer*4	Radius in plotter units.
ANGLE1	Real	Angle in degrees at which to begin drawing circle or arc.
ANGLE2	Real	Angle in degrees at which to terminate drawing circle or arc. (See Note 2.)

Notes: (1) An angle of 0° corresponds to the +x axis direction.
(2) If ANGLE1=ANGLE2, a circle will be drawn.
(3) If ANGLE1<ANGLE2, the arc will be plotted in a counterclockwise direction; if ANGLE1>ANGLE2, the arc will be plotted in a clockwise direction.

DASHL(IDASH,NELE,LPATRN,IFIX)

Set line mode to draw dashed lines using the pattern determined by the IDASH matrix and the length, LPATRN. If a "best fit" of the pattern is desired, set IFIX to indicate variable dashed lines.

Argument	Format	Description
IDASH	Integer Array	Two-dimensional array containing the dash/space pattern in relative units. IDASH (J,1) contains the relative length; IDASH (J,2) contains a 1 if the j^{th} element is a dash, a 2 if the j^{th} element is a space.
NELE	Integer	Number of elements forming the pattern (See Notes 2 and 3.)
LPATRN	Integer	Length in plotter units of the complete pattern. If LPATRN<0, solid line mode is restored. (See note 4.)
IFIX	=1 =2	If fixed dash line If variable dash line

Notes:
(1) A dash with length =0 is a dot in the pattern.
(2) IDASH should be dimensioned NELE x 2 in the callingprogram.
(3) A maximum of 8 dashes and 8 spaces are allowed to form a pattern.
(4) Dash mode is in effect until a return is explicitly made to solid line mode.
(5) In array order, all pattern lengths are followed by all corresponding element types.

DRAW(IX,IY)

Draw a vector from the current position with the pen down to the point with absolute coordinates (IX,IY).

Argument	Format	Description
IX,IY	Integer	x,y coordinates of endpoint in plotter units.

FNMBER(IX,IY,FNUM,ISW,NDEC)

Plot a floating point value in F format.

Argument	Format	Decription
IX,IY	Integer	x,y coordinates to move pen to prior to plotting number (absolute or incremental, depending on ISW).
FNUM	Real	Value of number to be plotted. (See note 1.)
ISW	Integer	0 => absolute pen move prior to plot. ≠0 => incremental pen move prior to plot
NDEC	Integer	Number of positions beyond decimal point to be plotted, where $0 \leq NDEC \leq 4$.

NOTES: (1) This routine is currently limited to values $< 10^{-4}$ or $> 3.2 \times 10^{5}$.

FONT(ISTD,IALT,ISEL)

Choose a standard and alternate font from the six available fonts (see ref(1)), and select the font which is to be currently in use.

Argument	Format	Description
ISTD	Integer ϵ[0,5]	Standard font
IALT	Integer ϵ[0,5]	Alternate font
ISTD	=1	To select standard font
	=2	To select alternate font

Notes: (1) <PLOTS> sets the standard font to 0 and the alternate font to 5.

(2) Font #5 includes a set of graphic symbols which are centered. These are very convenient for graphs in which points are to be denoted by symbols. These symbols have decimal codes 65 through 79. They are drawn starting from the center and the current pen location will not be altered.

GRIDSZ(IXMAX,IYMAX)

Divide the plotting grid into plotter units. The minimum plot increment is determined by the relationship between the graphic limits specified in <PLOTS> and IXMAX,IYMAX.

Argument	Format	Description
IXMAX,IYMAX	Integer	x,y coordinates of upper right corner of grid in plotter units

Notes: (1) The GRIDSZ routine indicates the number of grid divisions into which the paper size, as determined in PLOTS, is divided. It may be compared to laying a grid template over the usable portion of paper. The GRIDSZ divisions determine the minimum vector size for the plot. Thus, the number of divisions provides the resolution of the vectors.

(2) A combination of graphic limits and grid size which does not result in the same physical unit size on both axes will result in the drawing of an ellipse when a circle is desired, and vectors and characters with altered proportions.

(3) The default grid size is 3040 x 2000 plotter units.

(4) Approximately 200 machine units (.025mm) per inch can be distinguished by the eye.

ex: CALL GRIDSZ(100,100) divides the plot area determined in PLOTS into 100 plotter units in each direction.

IDRAW(IDELX,IDELY,N)

Draw incrementally from the current pen postion for N increments, with the pen down.

Argument	Format	Description
IDELX	Integer	Relative move in plotter units in the
	Array	x-direction for each vector to be drawn.
IDELY	Integer	Relative move in plotter units in the
	Array	y-direction for each vector to be drawn.
N	Integer	Number of vectors to be drawn.

```
ex: DIMENSION IDX(4), IDY(4)
    CALL IDRAW (IDX,IDY,4)

                where  IDX = {10,0,-10,0}
                and    IDY = {0,10,0,-10}
```

will draw a square 10 plotter units long.

IMOVE(IDELX,IDELY,N)

Move relative to the current pen position to the first pair of incremental values given and then draw incrementally the remaining N-1 pairs. (See <IDRAW>).

Argument	Format	Description
IDELX	Integer Array	Relative move in plotter units in the x-direction for each vector to be drawn.
IDELY	Integer Array	Relative move in plotter units in the y-direction for each vector to be drawn.
N	Integer	Number of vectors to be drawn.

LABEL(STRING,N)

Plot a string of characters at the current angle of rotation.

Argument	Format	Description
STRING	Byte Array	Array containing string of ASCII characters
N	Integer	Number of characters to plot. (See note 1.)

Notes: (1) If the contents of the string include the current label terminator character, the remaining characters in the string will not be plotted. See <LABEND> for discussion of terminators.

(2) After the call, the pen will be left in the appropriate position to plot another character.

ex: CALL LABEL ('ABC',3)

LABEND(LTERM)

Change the string terminator character for labels.

Argument	Format	Description
LTERM	Integer	Decimal value representing ASCII character of desired string terminator.

Notes: (1) The ETX (03) character is the default terminator.

(2) Choose a non-printing terminator character (01-37) to avoid visible print of the character at the end of the label.

(3) The NUL (00), ENQ (05), ESC (33), and DEL (127) are not allowed as terminators.

ex: CALL LABEND (13)

will change string terminator to a carriage return character.

LABSIZ(IHOR,IVERT)

Change the label size from current value to IHOR units horizontal size and IVERT units vertical size.

Argument	Format	Description
IHOR	Integer	Horizontal character size (including spacing) in plotter units. (See note 2.)
IVERT	Integer	Vertical character size in plotter units.

Notes: (1) The default values are IHOR=25, IVERT=25.

(2) The spacing automatically included with IHOR is calculated such that the actual character fills up (2/3) *IHOR.

(3) If the grid is not in equidistant plotter units in both x and y directions, characters will be plotted in a proportional manner to grid units.

LABSLT(SLANT)

Determine the slant angle to be used for printing labels.

Argument	Format	Description
SLANT	Real	Slant angle in degrees $\epsilon[0,180]$.

Notes: (1) The default slant angle is 90°. It is measured in a counterclockwise direction from the +x axis.

(2) The slant angle remains in effect until explicitly changed.

MOVE(IX,IY)

Move with the pen up to the point with absolute coordinates (IX,IY).

Arguments	Format	Description
IX,IY	Integer	x,y position of vector endpoint in plotter units.

NUMBER(IX,IY,NUM,INC)

Move the pen to (IX,IY) and print the integer NUM. If INC≠0, move the pen incrementally by (IX,IY) from the current position. Otherwise move absolutely to (IX,IY).

Argument	Format	Description
IX,IY	Integer	(x,y) coordinates (absolute if INC=0; incremental if INC≠0) of lower left plotting position of first digit in the number.
NUM	Integer	Integer in the range [-32767,+32767]
INC	=0	To move absolutely to (IX,IY)
	≠0	To move incrementally by (IX,IY)

PLOTD(IADV)

Current plot is complete. Plot paper is advanced and/or cut as indicated by the value of IADV. Pen is returned to holder, and plotter is disengaged.

Argument	Format	Description
IADV	Integer ϵ [0,5]	0 = Cutter disabled, no page advance 1 = Cutter disabled, full page advance 2 = Cutter disabled, half page advance 3 = Cutter enabled, full page advance 4 = Cutter enabled, half page advance

PLOTS(LUN,IERR,LOWX,LOWY,IUPX,IUPY)

Enable the plotter, determine graphic limits, initialize the plotter conditions for the next plot, and determine the I/O logical devices. This routine should be called before any other plotter routine.

Argument	Format	Description
LUN	Integer	Logical unit number for output to plotter.
IERR	Integer	Logical unit number for error and plot command code prints. If IERR = 0, no commands or error message printouts will occur. (See note 2.)
LOWX, LOWY, IUPX, IUPY	Integer	x and y coordinates of lower left and upper right graphic limits, respectively, in machine units. (See note 1.) If LOWX<0 the default machine units will be used.

Notes: (1) Graphic limits determine usable paper size. Machine units (.025mm) are required as the limit arguments. See section I.B. for choosing suitable values for the coordinates.

(2) All error messages and ASCII codes from HPLOT will appear in the file determined by IERR. It is recommended that IERR>0 while testing and IERR=0 for production runs.

ex: A paper size approximately 10"x10" may be established by the call:

```
CALL PLOTS (4,1,500,500,10660,10660)
```

Note that logical unit 4 will be assigned to the plotter, and logical unit 1 to print plot commands and error messages with this call.

ROTATE(ANGLE)

Rotate incremental vectors and characters drawn in label mode by ANGLE degrees.

Argument	Format	Description
ANGLE	Real	Angle of rotation in degrees.

Notes: (1) The default value of ANGLE = 0°.

(2) Absolute vectors are not affected by ROTATE.

SELPEN(NPEN)

Select a pen from those currently in the pen holders.

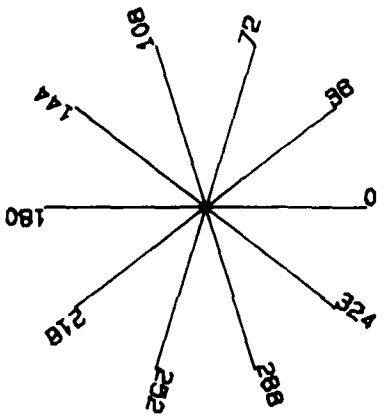
Argument	Format	Description
NPEN	Integer ϵ[1,8]	Number of the pen holder.

SECTION 3

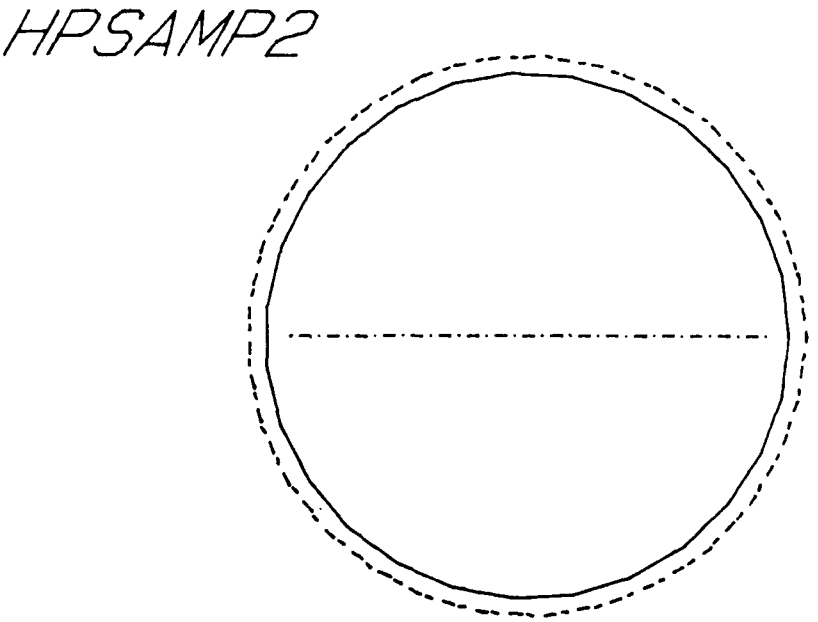
SAMPLE PROGRAMS

```
C HPSAMP1
C  PLOTTER SAMPLE SHOWING ROTATION,LABELLING
C
C
C ESTABLISH PLOTTING AREA OF APPROXIMATELY 3" x 4"
C     1"=1016 PLOTTER UNITS
          CALL PLOTS(4,0,1500,1500,4048,5064)
C 100 PLOTTER UNITS PER INCH
          CALL GRIDSZ(300,400)
          CALL LABSIZ(15,20)
          CALL MOVE(0,370)
          CALL LABEL('HPSAMP1.',8)
          CALL LABSIZ(10,10)
          DO 50 I=1,10
          CALL MOVE(150,150)
          ANGLE=36.*(I-1)
          CALL ROTATE(ANGLE)
          CALL IDRAW(110,0,1)
          NANGLE=ANGLE
          CALL NUMBER(0,2,NANGLE,1)
50        CONTINUE
          CALL PLOTD(0)
          STOP
          END
```

HPSAMP1.



```
C HPSAMP2
C  PLOTTER SAMPLE DEMONSTRATING CIRCLES,DASHED LINES,INCREMENTAL MOVE
C
      DIMENSION IDASH(4,2),IDASH2(4,2)
      DATA IDASH/1,2,2,1,     1,2,1,2/
      DATA IDASH2/1,0,1,1,    2,1,2,1/
C
C SET UP PLOT AREA 5" x 5"
      CALL PLOTS(1,2,2000,2000,7080,7080)
C SET UP GRID SIZE WITH 100 PLOTTER UNITS PER INCH
      CALL GRIDSZ(500,500)
      CALL MOVE(0,410)
      CALL LABSLT(60.)
      CALL LABEL('HPSAMP2.',8)
      CALL MOVE(450,250)
      CALL CIRCLE(150,0.,0.)
      CALL DASHL(IDASH,4,20,1)
      CALL MOVE(460,250)
      CALL CIRCLE(160,0.,0.)
      CALL IMOVE(-300,0,1)
      CALL DASHL(IDASH2,4,16,2)
      CALL IDRAW(280,0,1)
C
C TERMINATE DASHED LINE MODE
      CALL DASHL(IDASH,4,-1,0)
      CALL PLOTD(2)
      STOP
      END
```



SECTION 4

ADDING ROUTINES TO HPLOT

The HPLOT library is a two-layer set of FORTRAN subroutines. The upper layer contains those routines called directly by the programmer. The lower layer contains the set of routines called by the upper layer routines.

Any routines added to the library should use the lower layer routines to provide the actual commands and parameters in encoded form to the HP7221T. Appendix E of Hewlett Packard manual indicates the command format to be used for each plotter instruction. The five format routines will handle the encoding.

The named COMMON described in Appendix B must be included if necessary.

The lower layer routines are:

PSEND - Send plotter command and arguments as ASCII bytes
TRANS - Translate hexadecimal code to ASCII
MBA, MBP, PMB, SBN, MBN - The 5 different format types for command parameters

APPENDIX A

HPLOT ROUTINES

1. Routines Called Directly by User

```
C HPLOT.FTN            2/16/83
C
C   UTILITY ROUTINES TO DRIVE THE HP7221T PLOTTER
C   THESE ROUTINES CALL ON A SET OF LOWER LEVEL FORMAT ROUTINES IN
C    FORMATS.FTN , A HEX/ASCII TRANSLATION ROUTINE IN TRANS.FTN ,
C   AND THE I/O ROUTINE IN PSEND.FTN .
C-----------------------------------------------------------------------
C PLOTS -- INITIALIZE PLOTTER
C
      SUBROUTINE PLOTS(LUN,IERR,LOWX,LOWY,UPX ,UPY)
C FSD:11/19/82
      COMMON /HP7221/IOUNIT,IRUNIT,ETX
      BYTE IBYT(5),ETX(2)
      INTEGER UPX,UPY
      DATA ETX/03,0/
C
      IOUNIT=LUN
      IRUNIT=IERR
C DO THE LOGICAL UNIT ASSIGNMENT
      CALL ASSIGN(LUN,'TT2:',4)
C START PLOTTER
      IBYT(1)=27
      IBYT(2)=
      IBYT(3)=
      CALL PSEND(IBYT,3)
C INITIALIZE -- RETURN TO DEFAULT PLOT CONDITIONS
      CALL PSEND(
C SET UP DEVICE CONTROL:PLOTTER OUTPUT MODE, HANDSHAKING
      CALL PSEND(27,1)
      CALL PSEND('.M200;0;0;13;0: ,15)
      CALL PSEND(27,1)
      CALL PSEND('.N0;19:',7) !dc3 -- xoff
      CALL PSEND(27,1)
      CALL PSEND('.I160;0;17:',11) !dc1 -- xon;160 byte buffer trigger
C SET UP FOR DEBUG DEVICE TO BE ATTACHED
CCCC  CALL PSEND(27,1)
CCCC  CALL PSEND('.@;1:',5)!will work if baudrate=1200
C SET PEN TO 1
      CALL PSEND('vA',2)
C SET GRAPHIC LIMITS
      CALL PSEND('*W',2)
C SET LOWX=0 IF DEFAULT LIMITS ARE TO BE USED
      IF(LOWX.LT.0) GOTO 10
      CALL MBP(LOWX,LOWY,NBYTES,IBYT)
      CALL PSEND(IBYT,NBYTES)
      CALL MBP(UPX,UPY,NBYTES,IBYT)
      CALL PSEND(IBYT,NBYTES)
10    CALL PSEND('}',1)
C SET ALTERNATE FONT TO #5, THE GRAPHIC CHARACTER SET
      CALL FONT(0,5,1)
      RETURN
      END
C MOVE -- MOVE TO X,Y WITH PEN UP
      SUBROUTINE MOVE(IX,IY)
C FSD:9/20/82
C
      BYTE IBYT(6)
C
      IBYT(1)='P'
      CALL MBP(IX,IY,NBYTES,IBYT(2))
```

```
      CALL PSEND(IBYT,NBYTES+1)
      CALL PSEND(')',1)
      RETURN
      END
C DRAW -- MOVE TO X,Y WITH PEN DOWN
      SUBROUTINE DRAW(IX,IY)
C FSD:9/20/82
C
      BYTE IBYT(6)
C
      IBYT(1)='a
      CALL MBF(IX,IY,NBYTES,IBYT(2))
      CALL PSEND(IBYT,NBYTES+1)
      CALL PSEND(')',1)
      RETURN
      END
C PLOTD -- PLOT IS DONE
      SUBROUTINE PLOTD(IADV)
C FSD:9/22/82
C
C    IADV    0 CUTTER DISABLED, NO PAGE ADVANCE
C            1 CUTTER DISABLED, FULL PAGE ADVANCE
C            2 CUTTER DISABLED, HALF PAGE ADVANCE
C            3 CUTTER ENABLED,  FULL PAGE ADVANCE
C            4 CUTTER ENABLED,  HALF PAGE ADVANCE
C
      COMMON /HP7221/LUN,IERR,ETX
      BYTE IBYT(3),ETX(2)
C
C PUT PEN BACK IN STALL
      CALL SELPEN(0)
C ADVANCE AND CUT PAPER
      IF(IADV-1)20,11,10
10    IF(IADV-3)12,13,14
11    CALL PSEND('^)^+',4)
      GOTO 20
12    CALL PSEND('^)^-',4)
      GOTO 20
13    CALL PSEND('^(^+ ',4)
      GOTO 20
14    CALL PSEND('^(^-',4)
C TURN PLOTTER OFF
20    IBYT(1)=27
      IBYT(2)='.'
      IBYT(3)=')'
      CALL PSEND(IBYT,3)
C CLOSE LOGICAL UNIT LUN
      CLOSE(UNIT=LUN)
      RETURN
      END
C SELPEN -- SELECT PEN
      SUBROUTINE SELPEN(NPEN)
C FSD:9/20/82
C
      BYTE IBYT(3)
C
      IBYT(1)='v'
      CALL SBN(NPEN,IBYT(2))
      IBYT(3)='}'
      CALL PSEND(IBYT,3)
      RETURN
      END
C LABEL -- LABEL MODE ON AND PLOT GIVEN CHARACTER STRING
      SUBROUTINE LABEL(STRING,N)
C
C FSD:9/23/82
```

```
C
C INPUT IS THE CHARACTER STRING PLUS THE COUNT OF CHARACTERS(N)
C     THE COUNT WILL BE IGNORED IF THE TERMINATOR APPEARS IN THE STRING.
C     THE TERMINATOR IS THE ETX(=3) CHARACTER BY DEFAULT. TO CHANGE THE
C     TERMINATOR, USE THE SUBROUTINE  LABEND .
C
      COMMON /HP7221/LUN,IERR,ETX
      BYTE LB(2),STRING(1),ETX(2)
C
      LB(1)='^'
      LB(2)=39
      CALL PSEND(LB,2)
      DO 5 I=1,N
      IF(STRING(I).EQ.ETX(1)) GOTO 10
5     CONTINUE
      CALL PSEND(STRING,N)
      CALL PSEND(ETX(1),1)
      RETURN
10    CALL PSEND(STRING,I)
      RETURN
      END
C LABEND -- SET STRING TERMINATOR FOR LABELS
      SUBROUTINE LABEND(LTERM)
C FSD:9/22/82
C
C INPUT IS THE DESIRED TERMINATOR FOR SUBSEQUENT STRINGS
C
      COMMON /HP7221/LUN,IERR,ETX
      BYTE LB(2),ETX(2)
C
C NOTES:
C   (1) THE ETX CHARACTER IS THE DEFAULT STRING TERMINATOR
C   (2) ETX(1) CONTAINS TERMINATOR IN ASCII. A DUMMY POSITION IS
C       SAVED IN ETX TO RETAIN WORD BOUNDARIES IN COMMON.
C   (3) THE STRING TERMINATOR IS SENT TO PLOTTER. CHOOSE NON-PRINTING
C       CHARACTERS TO AVOID VISIBLE PRINT OF TERMINATOR!!
C
      ETX(1)=LTERM
      CALL PSEND('^\',2)
      CALL MBN(LTERM,NBYTES,LB)
      CALL PSEND(LB,NBYTES)
      RETURN
      END
C  LABSLT -- LABEL SLANT ANGLE
      SUBROUTINE LABSLT(SLANT)
C FSD:9/22/82
C
C INPUT IS THE LABEL SLANT ANGLE IN DEGREES
C DEFAULT SLANT ANGLE IS 90 DEGREES
C ANGLES >180 ARE TREATED AS 180 DEGREES LESS THAN GIVEN VALUE
C
      BYTE LB(3)
C
      CALL PSEND('^/',2)
      CALL MBA(SLANT,NBYTES,LB)
      CALL PSEND(LB,NBYTES)
      RETURN
      END
C IDRAW -- INCREMENTAL DRAW
      SUBROUTINE IDRAW(IDELX,IDELY,N)
C FSD:9/22/82
C
      COMMON /HP7221/LUN,IERR,EXT
      BYTE EXT(2),IBYTES(6)
      DIMENSION IDELX(1),IDELY(1)
C
```

```
      CALL PSEND('s',1)
      DO 50 I=1,N
      CALL PMB(IDELX(I),IDELY(I),NBYTES,IBYTES)
      CALL PSEND(IBYTES,NBYTES)
50    CONTINUE
      CALL PSEND('e',1)
      RETURN
      END
C IMOVE -- INCREMENTAL MOVE AND THEN DRAW
      SUBROUTINE IMOVE(IDELX,IDELY,N)
C FSD:9/22/82
C
      COMMON /HP7221/LUN,IERR,EXT
      BYTE EXT(2),IBYTES(6)
      DIMENSION IDELX(1),IDELY(1)
C
      CALL PSEND('m',1)
      DO 50 I=1,N
      CALL PMB(IDELX(I),IDELY(I),NBYTES,IBYTES)
      CALL PSEND(IBYTES,NBYTES)
50    CONTINUE
      CALL PSEND('e',1)
      RETURN
      END
C GRIDSZ -- SET GRID SIZE
      SUBROUTINE GRIDSZ(IXMAX,IYMAX)
C FSD: 9/23/82
C
C  SET GRID SIZE IN PLOTTER UNITS
      BYTE NP(5)
C
      CALL PSEND('*S',2)
      CALL MBP(IXMAX,IYMAX,NBYTES,NP)
      CALL PSEND(NP,NBYTES)
      CALL PSEND('e',1)
      RETURN
      END
C DASHL -- DRAW FIXED OR VARIABLE DASHED LINES
      SUBROUTINE DASHL(IDASH,NELE,LPATRN,IFIX)
C FSD: 9/30/82
C
C      REVISED 4/1/83
C
C
C INPUTS:
C     IDASH - TWO-DIMENSIONAL ARRAY CONTAINING DASH/SPACE PATTERN,
C         WHERE THE NUMBER OF UNITS FOR THE JTH ELEMENT IS IN IDASH(J,1)
C           AND IDASH(J,2) = 1 --> DASH
C                            2 --> SPACE
C         **NOTE THAT THE NUMBER OF UNITS IS RELATIVE(FROM 0 TO 31)
C     NELE = NUMBER OF ELEMENTS IN THE PATTERN (MAXIMUM J IN IDASH)
C     LPATRN = PATTERN LENGTH (I.E.,REPEAT CYCLE) IN PLOTTER UNITS
C              IF LPATRN<0 DASH MODE IS TERMINATED
C     IFIX = 1 --> FIXED DASH LINE
C            2 --> VARIABLE DASH LINE
C NOTES:
C  (1) A DASH WITH LENGTH 0 IS A DOT IN THE PATTERN
C  (2) A MAXIMUM OF 8 DASHES AND 8 SPACES ALLOWED IN ANY PATTERN
C  (3) DASH MODE IS IN EFFECT UNTIL SPECIFICALLY TERMINATED BY A
C         CALL TO DASHL WITH LPATRN<0
C
      DIMENSION IDASH(NELE,2)
      BYTE CMD(2),IBYT(16)
      DATA CMD/'D','R'/
C
      CALL PSEND('~',1)
```

```
        IF=IFIX                 !FSD 9/1/83
        IF(IFIX.LT.1 .OR. IFIX.GT.2) IF=1
        CALL PSEND(CMD(IF),1)
        IF(LPATRN.LT.0) GOTO 20
        NSP=0
        NDA=0
        DO 10 N=1,NELE
        IF(IDASH(N,2).EQ.2) GO TO 5
C IF A DASH THEN USE MODIFIED SBN FORMAT (ADD 32)
        IARG=IDASH(N,1)+32
        NDA=NDA+1
        IF(NDA.GT.8) STOP 9
        GOTO 10
5       NSP=NSP+1
        IF(NSP.GT.8) STOP 9
        IARG=IDASH(N,1)
10      CALL SBN(IARG,IBYT(N))
        CALL PSEND(IBYT,NELE)
        CALL MBN(LPATRN,NBYTES,IBYT)
        CALL PSEND(IBYT,NBYTES)
20      CALL PSEND(')',1)
        RETURN
        END
C NUMBER -- CONVERT INTEGER*2 TO ASCII AND PLOT
        SUBROUTINE NUMBER(IX,IY,NUM,INC)
C FSD: 9/28/82
C
        BYTE ASCNUM(6)
C MOVE TO IX,IY PRIOR TO PLOTTING THE NUMBER
C IF INC#0 THEN MOVE INCREMENTALLY; OTHERWISE, ABSOLUTELY.
C
        CALL TRANS(NUM,ASCNUM,NBYTES)
        IF(INC.EQ.0) GOTO 10
        CALL IMOVE(IX,IY,1)
        GOTO 20
10      CALL MOVE(IX,IY)
20      CALL LABEL(ASCNUM,NBYTES)
        RETURN
        END
C FONT -- DEFINE STANDARD AND ALTERNATE FONTS FOR LABELS
        SUBROUTINE FONT(ISTD,IALT,ISEL)
C
C OCTOBER 20,1982:FSD
C
C THIS ROUTINES ALLOWS A CHANGE FROM THE DEFAULT STANDARD AND/OR ALTERNATE
C FONTS. IT ALSO DETERMINES WHICH OF THE TWO WILL BE IN EFFECT UNLESS CHANGED
C BY A CALL TO THIS ROUTINE.
C TO TEMPORARILY CHANGE FROM STANDARD TO ALTERNATE FONTS WHILE PLOTTING
C A LABEL, PROCEED AS FOLLOWS:
C    TO SELECT THE STANDARD FONT, INSERT THE SO(14) CHARACTER IN THE CHARACTER
C    STRING PRIOR TO THE CHARACTERS TO BE PLOTTED.
C    TO SELECT THE ALTERNATE FONT, USE THE SI(15) CHARACTER IN THE SAME WAY.
C DEFAULT FONTS ARE 0,0
C
C       ISTD= STANDARD FONT
C       IALT= ALTERNATE FONT
C       ISEL= SELECT FONT TO BE IN EFFECT(1=>STANDARD, 2=>ALTERNATE)
C
C NOTE: GRAPHIC FONT#5 DOES NOT USE THE AUTOMATIC HORIZONTAL SPACE FEATURE
C       FOR THE SPECIAL GRAPHIC CHARACTERS REPRESENTED BY DEC. CODES 64-79
C
        COMMON /HP7221/LUN,IER,ETX
        BYTE BYT(5),CMD(2),ETX(2)
        DATA CMD/15,14/
C
        CALL PSEND('^P',2)
```

```
        CALL MBP(ISID,IAL1,NBYTES,BYT)
        CALL PSEND(BYT,NBYTES)
        CALL PSEND('^',2)
        CALL PSEND(CMD(ISEL),1)
        CALL PSEND(ETX,1)
        RETURN
        END
C
C LABSIZ -- CHANGE SIZE OF LABEL
        SUBROUTINE LABSIZ(IHOR,IVER)
C FSD: 10/6/82
C
C EITHER VERTICAL OR HORIZONTAL SIZE MAY BE ALTERED INDEPENDENTLY
C DEFAULT = 25 PLOTTER UNITS VERTICAL, 25 PLOTTER UNITS HORIZONTAL
C TO RETURN TO DEFAULT SIZE, SET IHOR<=0
C SIZE REMAINS UNTIL EXPLICITLY ALTERED
C
        CALL PSEND('^Z',2)
        IF(IHOR.LE.0) GOTO 10
        LINE=2*IVER
        CALL MBP(IHOR,LINE,NBYTES,BYT)
10      CALL PSEND(BYT,NBYTES)
        RETURN
        END
C ROTATE -- ROTATE INCREMENTAL VECTORS
        SUBROUTINE ROTATE(ANGLE)
C FSD: 1/4/83
C ROTATE ALL SUBSEQUENT INCREMENTAL VECTORS BY ANGLE DEGREES
C
        BYTE STR(3)
C
C RESTORE ROTATION ANGLE TO 0
        CALL PSEND('w)',2)
C ROTATE BY ANGLE DEGREES
        CALL PSEND('w',1)
        CALL MBA(ANGLE,NBYT,STR)
        CALL PSEND(STR,NBYT)
        RETURN
        END
C FNMBER -- PLOT A FLOATING POINT NUMBER

        SUBROUTINE FNMBER(IX,IY,FNUM,ISW,NDEC)
C FSD:1/6/83
C PLOTS A FLOATING POINT NUMBER IN F FORMAT;I.E.,XXX.XX
C LIMITATIONS:
C       (1)  NOT USEFUL FOR VERY SMALL OR VERY LARGE NUMBERS
C
C   IX,IY = POSITION TO MOVE PEN TO, EITHER INCREMENTAL OR ABSOLUTE
C   FNUM  = THE FLOATING POINT NUMBER TO PLOT
C   ISW   = 0 FOR ABSOLUTE MOVE TO IX,IY; #0 FOR INCREMENTAL MOVE
C   NDEC  = NUMBER OF POSITIONS BEYOND THE DECIMAL POINT TO PLOT
C           **MAXIMUM NDEC = 4**
C NUMBERS ARE ROUNDED ACCORDING TO NDEC
C
        BYTE ASCII(20)
C
        IF(ISW.EQ.0) GOTO 10
        CALL IMOVE(IX,IY,1)
        GOTO 20
10      CALL MOVE(IX,IY)
20      NUM=FNUM
        IF(NDEC.GT.0) GOTO 21
        CALL TRANS(NUM,ASCII,N)
        CALL LABEL(ASCII,N)
        RETURN
```

```
C
21      IF(NDEC.GT.4)NDEC=4   ! because TRANS uses a 2-byte integer input
        IPOW10=10**NDEC
        FRAC=ABS(FNUM-NUM)*IPOW10
        IFRAC=FRAC
        IF(FRAC-IFRAC.LT..5) GOTO 25
        IFRAC=IFRAC+1
        IF(IFRAC.LT.IPOW10) GOTO 25
        NUM=(IABS(NUM)+1)*ISIGN(1,NUM)
        IFRAC=0
25      CALL TRANS(NUM,ASCII,N)
        CALL LABEL(ASCII,N)
        CALL LABEL('.',1)
        CALL TRANS(IFRAC,ASCII,N)
        IF(N.EQ.NDEC) GOTO 30
        NZ=NDEC-N
        DO 28 I=N,1,-1
28      ASCII(I+NZ)=ASCII(I)
        DO 29 I=1,NZ
29      ASCII(I)=48
30      CALL LABEL(ASCII,NDEC)
        RETURN
        END
C ADVANC -- ADVANCE PAGE
        SUBROUTINE ADVANC(IADV)
C ADVANC -- ADVANCE AND/OR CUT PAGE
C           AND SET UP PLOTTER AREA FOR NEXT PLOT
C  FSD:2/03/83
C
C    IADV = 0 CUTTER DISABLED, NO PAGE ADVANCE
C           1 CUTTER DISABLED, FULL PAGE ADVANCE
C           2 CUTTER DISABLED, HALF PAGE ADVANCE
C           3 CUTTER ENABLED,  FULL PAGE ADVANCE
C           4 CUTTER ENABLED,  HALF PAGE ADVANCE
C
        COMMON /HP7221/LUN,IERR,ETX
        BYTE IBYT(5),ETX(2)
C
C ADVANCE AND CUT PAPER
        IF(IADV-1)20,11,10
10      IF(IADV-3)12,13,14
11      CALL PSEND('^)^+',4)
        GOTO 20
12      CALL PSEND('^)^-',4)
        GOTO 20
13      CALL PSEND('^(^+',4)
        GOTO 20
14      CALL PSEND('^(^-',4)
20      RETURN
        END
C CIRCLE -- PLOT CIRCLE OR ARC
        SUBROUTINE CIRCLE(IRAD,ANGLE1,ANGLE2)
C FSD:9/23/82
C    REVISED 1/31/83 (FSD)
C IT NEEDS A MORE GENERAL TECHNIQUE OF HANDLING LG RADII STEP SIZING****
C  RIGHT NOW, IT ASSUMES P.U. AT 200/INCH
C
        INTEGER*4 IRAD
        BYTE NR(3),NS(3),INS(2)
        DATA INS/'u','t'/,  RADN/.0174532925/
C
C     IRAD = RADIUS IN PLOTTER UNITS
C     ANGLE1 = STARTING ANGLE IN DEGREES
C     ANGLE2 = STOP ANGLE IN DEGREES
C THE ARC WILL BE PLOTTED
C                 COUNTERCLOCKWISE DIRECTION IF ANGLE1<ANGLE2
C                 CLOCKWISE DIRECTION IF ANGLE1>ANGLE2
```

```
C IF ANGLE1=ANGLE2, A CIRCLE WILL BE DRAWN STARTING AT THE GIVEN ANGLE
C THE PLOTTER COMMAND MAY BE USED IF IRAD.32768.
C OTHERWISE, AN ALGORITHM IS USED TO PLOT LINE SEGMENTS.
C
      I=1
      IF(ANGLE1.GT.ANGLE2) I=2
      IF(IRAD.LT.0) GOTO 50
      IF(IRAD.EQ.0) RETURN
      IF(IRAD.GT.32767) GOTO 40
      IRD=IRAD !mbn routine needs integer*2 argument
      CALL PSEND(INS(I),1)
      CALL MBN(IRD,NBYTES,NR)
      CALL PSEND(NR,NBYTES)
      CALL MBA(ANGLE1,NBYT,NS)
      CALL PSEND(NS,NBYT)
      IF(ANGLE1.EQ.ANGLE2) GOTO 30
      CALL MBA(ANGLE2,NBYT,NS)
      CALL PSEND(NS,NBYT)
30    CALL PSEND('>',1)
      RETURN
C FOR LARGE RADII, NEED ALGORITHM:
C  SINCE FOR LARGE RADIUS, GET A RATHER FLAT CURVE, CAN USE A LINE SEGMENT
C  ENCOMPASSING A MULTIPLE OF THE  PLOTTER UNIT SIZE
40    DELTHT=50./IRAD
      IF(ANGLE1.NE.ANGLE2) GOTO 41
      THETA=ANGLE1*RADN
      THETAN=(ANGLE1+360.)*RADN
      N=360.*(RADN/DELTHT) + 1
      GOTO 42
41    IF(I.EQ.2)DELTHT=-DELTHT
      N=(ANGLE2-ANGLE1)*(RADN/DELTHT) + 1
      THETA=ANGLE1*RADN
      THETAN=ANGLE2*RADN
42    CTHT=COS(THETA)
      STHT=SIN(THETA)
      DXADD=0.
      DYADD=0.
      DO 45 K=1,N-1
      ARG=THETA+K*DELTHT
      CTHT1=COS(ARG)
      STHT1=SIN(ARG)
      DX=IRAD*(CTHT1-CTHT)
      DY=IRAD*(STHT1-STHT)
      IDX=DX+SIGN(.5,DX)
      IDY=DY+SIGN(.5,DY)
      CTHT=CTHT1
      STHT=STHT1
      DXADD=DXADD+(DX-IDX) !keep track of cumulative errors
      IF(ABS(DXADD).LT.1.) GOTO 43
      IDX=IDX+SIGN(1.,DXADD)
      DXADD=DXADD-SIGN(1.,DXADD)
43    DYADD=DYADD+(DY-IDY)
      IF(ABS(DYADD).LT.1.) GOTO 44
      SGN=SIGN(1.,DYADD)
      IDY=IDY+SGN
      DYADD=DYADD-SGN
44    IF(IDX.EQ.0 .AND. IDY.EQ.0) GOTO 45
      CALL IDRAW(IDX,IDY,1)
45    CONTINUE
      CTHT1=COS(THETAN)
      STHT1=SIN(THETAN)
      DX=IRAD*(CTHT1-CTHT)
      DY=IRAD*(STHT1-STHT)
      IDX=DX+SIGN(.5,DX)
      IDY=DY+SIGN(.5,DY)
      DXADD=DXADD+(DX-IDX)
      DYADD=DYADD+(DY-IDY)
      IDX=IDX+DXADD+SIGN(.5,DXADD)
      IDY=IDY+DYADD+SIGN(.5,DYADD)
      CALL IDRAW(IDX,IDY,1)
      RETURN
50    STOP 'CIRCLE ERROR, RADIUS<0.'
      END
```

2. Routines Called Internally From HPLOT

```
      SUBROUTINE PSEND(IBYT,NBYTES)
C FSD:  9/82
      COMMON /HP7221/LUN,IERR,ETX
      BYTE IBYT(1)
      DIMENSION IOSB(2),IOPAR(6)
C
      IOPAR(2)=NBYTES
      IOPAR(3)=0
      CALL GETADR(IOPAR(1),IBYT(1))
      CALL WTQIO('000410,LUN,1,,IOSB,IOPAR,IDS)
      IF(IERR.LE.0)GOTO 80
      WRITE(IERR,105) NBYTES,(IBYT(K),K=1,NBYTES)
      WRITE(IERR,106) (IBYT(K),K=1,NBYTES)
80    IF(IDS.LT.0) GOTO 90
      IF(IOSB(1).LT.0) GOTO 95
      RETURN
90    WRITE(IERR,100) IDS
      STOP 1
95    WRITE(IERR,101) IOSB(1)
      STOP 2
C
100   FORMAT(' PSEND:QIO Directive Error',I5)
101   FORMAT(' PSEND:QIO Write I/O Error',I5)
105   FORMAT(I5,'///',10I5)
106   FORMAT(9X,10A1)
      END
```

```
C FORMATS.FTN          2/16/83
C
C  THESE ROUTINES PROVIDE THE ALGORITHMS FOR HANDLING ALL POSSIBLE ARGUMENT
C  TYPES NEEDED FOR THE HP7221 COMMAND LANGUAGE.
C
C  UPDATED ON 3/2/83 TO FIX UP ERROR MESSAGES,STOP STATEMENTS, ETC.
C-------------------------------------------------------------------------
C MBP -- MULTIPLE byte PAIR PARAMETER FORMAT
      SUBROUTINE MBP(NX,NY,NBYTES,NP)
      COMMON /HP7221/LUN,IERR
      BYTE NP(5)
C
      IF(NX.LT.0 .OR. NY.LT.0) GOTO 5
      N=NX
      IF(NX.LE.NY)N=NY
      IF(N.LT.256)GOTO 10
      IF(N.LT.2048)GOTO 14
      IF(N.LT.16384)GOTO 15
5     WRITE (IERR,990) NX,NY
990   FORMAT(' MBP: VALUE OUT OF RANGE.  NX=',I8,'   ,NY=',I8)
      STOP 5
10    IF(N.GT.31) GOTO 13
      IF(N.GT.3)GOTO 12
C ONE-BYTE FORMAT
      NP(1)=NY+96+4*NX
      NBYTES=1
      RETURN
C TWO-BYTE FORMAT
12    NX1=NX/2
      NX2=NX-2*NX1
      NP(1)=NX1+96
      NP(2)=NY+32*NX2
      NBYTES=2
      GOTO 24
C THREE-BYTE FORMAT
13    NX1=NX/16
      NX2=NX-16*NX1
      NY2=NY/64
      NY3=NY-64*NY2
      NP(1)=NX1+96
      NP(2)=NY2+4*NX2
      NP(3)=NY3
      NBYTES=3
      GOTO 23
C FOUR-BYTE FORMAT
14    NX1=NX/128
      NXR=NX-128*NX1
      NX2=NXR/2
      NX3=NXR-2*NX2
      NY3=NY/64
      NY4=NY-64*NY3
      NP(1)=96+NX1
      NP(2)=NX2
      NP(3)=NY3+32*NX3
      NP(4)=NY4
      NBYTES=4
      GOTO 22
C FIVE-BYTE FORMAT
15    NX1=NX/1024
      NXR=NX-1024*NX1
```

```
      NX2=NXR/16
      NX3=NXR-16*NX2
      NY3=NY/4096
      NYR=NY-4096*NY3
      NY4=NYR/64
      NY5=NYR-64*NY4
      NP(1)=96+NX1
      NP(2)=NX2
      NP(3)=NY3+4*NX3
      NP(4)=NY4
      NP(5)=NY5
      NBYTES=5
21    IF(NP(5).LE.31)NP(5)=NP(5)+64
22    IF(NP(4).LE.31)NP(4)=NP(4)+64
23    IF(NP(3).LE.31)NP(3)=NP(3)+64
24    IF(NP(2).LE.31)NP(2)=NP(2)+64
      RETURN
      END
C SBN -- SINGLE BINARY NUMBER FORMAT
      SUBROUTINE SBN(IARG,IBYT)
      BYTE IBYT
C
      IF (IARG.LT.0) STOP 3
      IBYT=IARG
      IF(IBYT.GT.31) RETURN
      IBYT=IBYT+64
      RETURN
      END
C MBA -- MULTIPLE-BYTE ANGLE FORMAT
      SUBROUTINE MBA(THETA,NBYTES,NP)
C
C FSD: 9/20/82
C
C     INPUT=THETA, AN ANGLE SPECIFIED IN DEGREES (REAL NUMBER).
C     IF THETA IS NOT IN RANGE [0,360), AN ANGLE IN THIS RANGE
C     IS USED.
C
      BYTE NP(3)
C
      ANGLE=THETA
1     IF(ANGLE.GE.0) GOTO 2
      ANGLE=ANGLE+360.
      GOTO 1
2     IF(ANGLE.LT.360.)GOTO 3
      ANGLE=AMOD(ANGLE,360.)
3     NP(1)=0
      IF(ANGLE.LT.180.) GOTO 10
      NP(1)=8
      ANGLE=ANGLE-180.
10    NA=(ANGLE/90.)*16384
      NA1=NA/4096
      NP(1)=NP(1)+NA1+96
      NR=NA-4096*NA1
      NBYTES=1
      IF(NR.EQ.0) RETURN
      NA2=NR/64
      NA3=NR-64*NA2
      IF(NA3.EQ.0) GOTO 20
      NBYTES=NBYTES+1
      NP(3)=NA3
      IF(NA3.LE.31)NP(3)=NP(3)+64
20    NP(2)=NA2
      IF(NA2.LE.31)NP(2)=NP(2)+64
      NBYTES=NBYTES+1
      RETURN
      END
```

```
C MBN -- MULTIPLE-BYTE NUMBER FORMAT
      SUBROUTINE MBN(IARG,NBYTES,NP)
C  FSD: 9/20/82
C
      COMMON /HP7221/LUN,IERR,ETX
      BYTE NP(3),ETX(2)
C
      IF(IARG.LT.0) GOTO 10
      IF(IARG.LT.16) GOTO 1
      IF(IARG.LT.1024) GOTO 2
      GOTO 3
10    IF(IERR.GT.0) WRITE(IERR,900)IARG
      STOP 8
C
C ONE-BYTE FORMAT
1     NBYTES=1
      NP(1)=IARG+96
      RETURN
C TWO-BYTE FORMAT
2     NBYTES=2
      NN1=IARG/64
      NP(2)=IARG-64*NN1
      GOTO 12
C THREE-BYTE FORMAT
3     NBYTES=3
      NN1=IARG/4096
      NR=IARG-4096*NN1
      NP(2)=NR/64
      NP(3)=NR-64*NP(2)
      IF(NP(3).LE.31) NP(3)=NP(3)+64
12    IF(NP(2).LE.31) NP(2)=NP(2)+64
      NP(1)=NN1+96
      RETURN
900   FORMAT(' *** ERROR IN MBN: ARG.OUT OF RANGE = ',I8)
      END
C PMB -- PAIR OF MULTIPLE BYTE NUMBERS
      SUBROUTINE PMB(N1,N2,NBYTES,NP)
C FSD:9/22/82
C
      COMMON /HP7221/LUN,IERR,ETX
      BYTE ETX(2),NP(6)
      DIMENSION NX(2)
C
C FOLLOW SAME PROCEDURE FOR X AND Y EXCEPT THAT FLAG CHANGES
      L=0
      N=0
      IFL=64
      NX(1)=N1
      NX(2)=N2
1     L=L+1
      IF(L-2)4,3,2
2     NBYTES=N
      RETURN
3     IFL=32
4     IF(NX(L).LT.0) GOTO 5
      IF(NX(L).LT.16) GOTO 11
      IF(NX(L).LT.512) GOTO 21
      IF(NX(L).LT.16384) GOTO 31
      IF(IERR.GT.0) WRITE(IERR,900) NX(L)
900   FORMAT(' *** ERROR IN PMB: NX OR NY OUT OF RANGE.',I8)
      STOP 6
C
5     IF(NX(L).GE.-16) GOTO 10
      IF(NX(L).GE.-512) GOTO 20
      IF(NX(L).GE.-16384) GOTO 30
      IF(IERR.GT.0) WRITE(IERR,900) NX(L
```

```
        STOP 6
C ONE-BYTE FORMAT
10      NX(L) NX(L)+32
11      K=K+1
        NP(K) NX(L)+1FL
        GOTO 1
C TWO-BYTE FORMAT
20      NX(L) NX(L)+1024
21      NX1=NX(L)/32
        NX2 NX(L)-32*NX1
        K=K+1
        NP(K) NX1+1FL
        K K+1
        NP(K) NX2+1FL
        GOTO 1
C THREE-BYTE FORMAT
30      NX(L) NX(L)+32768
31      NX1 NX(L)/1024
        NXR NX(L)-1024*NX1
        NX2=NXR/32
        NX3 NXR-32*NX2
        K K+1
        NP(K) NX1+1FL
        K=K+1
        NP(K)=NX2+1FL
        K K+1
        NP(K)=NX3+1FL
        GOTO 1
C
        END
```

```
C TRANS -- TRANSLATE INTEGER INTO PRINTABLE CHARACTERS
      SUBROUTINE TRANS(INT,ASC,NA)
C FSD:9/27/82
C
      BYTE ASC(1)
      INTEGER*2 INT
C INPUT IS A 16 BIT INTEGER IN THE RANGE [-32767,+32767]
C OUTPUT IS AN ARRAY OF ASCII BYTES 1 TO 6 ELEMENTS IN LENGTH
C
      ISW=0
      K=0
      NN1=INT
      IF(INT)10,5,20
C ARGUMENT IS 0
5     NA=1
      ASC(1)=48
      RETURN
C ARGUMENT IS NEGATIVE
10    ASC(1)='-'
      K=1
      NN1=-NN1
C ARGUMENT IS POSITIVE
20    DO 25 IEXP=4,0,-1
      IPTEN=10**IEXP
      NN2=NN1/IPTEN
      IF(NN2.EQ.0 .AND. ISW.EQ.0) GOTO 25
      K=K+1
      ISW=1
      ASC(K)=NN2+48
      NN1=NN1-NN2*IPTEN
25    CONTINUE
      NA=K
      RETURN
      END
```

APPENDIX B

ARGUMENTS IN COMMON AREA

All HPLOT routines reference the named COMMON area HP7221.

The program statement is:

```
COMMON/HP7221/LUN,IERR,ETX
```

where

```
LUN  = logical I/O unit for the plotter
IERR = logical I/O unit for error messages from HPLOT routines
ETX  = default string terminator in hi-order byte, lo-order byte
       used for padding only
```

APPENDIX C

ERROR MESSAGES

1. ERROR MESSAGES FROM HPLOT

Error Stop #	Originating Routine	Cause of Error
1	PSEND	QIO Directive Error
2	PSEND	QIO Write Error
3	SBN	Argument<0
5	MBP	Argument>16384 or <0
6	PMB	Argument out of range
8	MBN	Argument≥32768
9	DASHL	Number of dashes or spaces in dashed line pattern exceeds 8.
-	CIRCLE	Radius of circle cannot be negative.

2. ERROR MESSAGES FROM PLOTTER MICROPROCESSOR

When the HP7221T Error Light turns on, the plotter is ready to send an error message to the user. To see this meassage, do not turn the plotter off. Terminate or abort the run if still in progress and run ERDUMP, the program listed on the following page.

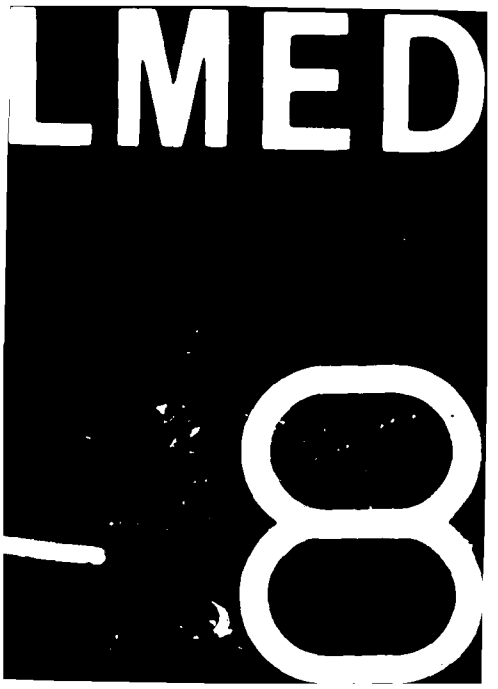
ERDUMP returns three parameters to the user terminal, as indicated below:

ITYPE = Error type of most recent error. This is an integer ϵ[0,99]. See the HP manual, Appendix C, for an explanation of the error types.

IBYTE = Instruction code or parameter which caused the error.

NERR = Number of errors since PLOTS was called.

```
C ERDUMP -- OUTPUT ERROR INFORMATION FROM PLOTTER
C  PSD:3/2/83
C
C
      COMMON /HP7221/IOUNIT,IRUNIT,ETX
C
      IOUNIT=1
      CALL ASSIGN(IOUNIT,'TT2: ',4)
      CALL PSEND('>',1)
      CALL PSEND(27,1)
      CALL PSEND('.(',2)      !be sure plotter is still on
      CALL PSEND(27,1)
      CALL PSEND('.E',2)
      READ (IOUNIT,*) ITYPE,IBYTE,NERR
      TYPE *,'ERROR TYPE=',ITYPE,'    PARAMETER=',IBYTE,
     $          '    # ERRORS=',NERR
      STOP
      END
      .
```


LMED
8